Attorney's Docket No.: 34874-020/2003P00061US01

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application Serial No.:		10/628,959	Conf. No.:	6174
Applicant	:	Eitan Hefetz		
Filed	:	July 28, 2003		
TC/Art Unit	:	2178		
Examiner	:	Manglesh M. Patel		
For	:	**SELECTIVELY INTERPRETED PORTAL PAGE LAYOUT TEMPLATE**		

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

DECLARATION UNDER 37 CFR § 1.131

1. This declaration is being made to establish a date of invention of at least December 18, 2002 for the subject matter defined in Claims 1-25 of the above-identified patent application.

2. The undersigned, Eitan Hefetz, Aviv Levin, Yossi Pik and Yaniv Haber are all current or former employees of a company affiliated with SAP AG (collectively referred to as "SAP").

3. The above-referenced patent application claims priority to U.S. Pat. App. Ser. No. 60/435,637, which was filed on December 20, 2002.

4. We hereby declare that, as of at least December 18, 2002, we invented a design-time translator and a run-time translator according to the systems, articles, and methods set forth in claims 1-25.

5. The attached SAP internal designed guides (attached hereto as Exhibits A and B), which are partially redacted, were generated and internally distributed on or before December

18, 2002 and demonstrate that the claimed subject matter was conceived and reduced to practice on or before December 18, 2002.

6. Exhibit A describes a WYSIWYG (what you see is what you get) portal page layout template module for use in connection with the SAP Enterprise Portal version 6.0 (as indicated by "EP6.0"). Exhibit A describes an arrangement in which placeholders are used to simulate run-time representations of content components at design-time. Layout template files are described that use dedicated tags to set the placeholders for containers which contain the content components. A run-time tag handler includes content components' run-time generated content while a design-time tag handler paints a schematic Accordingly, Exhibit A provides support for claims 1, 6, 10, 14, 18, and 21.

7. Exhibit B describe a page layout module used in connection with the SAP Enterprise Portal version 6.0. Exhibit B describes a page layout module in charge of constructing Portal Page HTML by retrieving iViews content from a PageBuilder module and merging it into a wrapping layout template HTML to provide a desired page look and content arrangement. In this document, a JSP Template defines a template for a page layout, a Page Builder module builds a page, fetches data and calls a Layout Component to construct the page, the Layout Component is a run-time engine for constructing page layout, a Main Content Storage component holds and manages data (content, tray) fetched or created by Page Builder, a Layout TagLib is a library in use within JSP Templates to serve Layout Component requirements such as placeholders for iViews Containers, iViews Containers are a page layout entity which contains iViews that should be presented together, in a certain order, on a page, and an ILayoutStructure is an interface for getting page layout structure as containers and iViews arrangements on the page. These modules and components are further described and illustrated through Exhibit B. Accordingly, Exhibit B provides support for claims 1, 6, 10, 14, 18, and 21.

8. We hereby declare that all statements made herein of our own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by a fine or imprisonment, or both, under Section 1001 of Title 18 of the

Attorney's Docket No.: 34874-020 /2003P00061US01

United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

[illegible]
Eitan Hefetz

Date: ~~June~~ July 27, 2008

____
Aviv Levin

Date: June ____, 2008

____
Yossi Pik

Date: June ____, 2008

____
Yaniv Haber

Date: June ____, 2008

EXHIBIT A

[redacted]

WYSIWYG portal page layout template

EP6.0

The Concept

Using Template files with place holders

The provided solution is based on layout template files.
Those files use dedicated tags to set place holders for containers which contain the content components.
At run-time the placeholders will be replaced with the actual components presenting their content while at design-time the placeholders will present content components representation to simulate the run-time presentation.
The page general structure is determined by using html tags (like html <table>).
The template files can contain any other html elements or additional code, which will be rendered and presented both at run-time and design-time.

One file for both run-time and design-time

Using the same layout template file for both run-time (presenting the actual page to the user) and design-time (editing the page content), provide create-once-use-twice solution and enables easy maintenance.
This architecture ensures presenting exactly same page appearance both at run-time and at design-time.

The Implementation

Jsp layout template file with dedicated tag library tags

The implementation uses jsp files as the layout templates. Such jsp file can contain any html, java, tag-lib or other jsp complying content, to be included in the portal page.
The content components containers place holders are marked with a dedicated tag library, using the <lyt:container> tag. Such tag, located in the jsp, marks a place for a column of content components (iViews).

Run-time and Design-time rendering

The provided solution uses different <lyt:container> tag handler implementations for run-time and design-time.
The run-time tag handler includes the content components' run-time generated content.
The design-time tag handler paints a schematic run-time-look-alike representation of each content component with code enabling edit the page (adding, removing, re-locating components).
All other jsp content is rendered the same.

EXHIBIT B

[redacted]

Page Layout – Run Time

EP6.0

Module Objective

The Page Layout Module is in charge of constructing the full Portal Page HTML by retrieving the iViews content from the Page Builder and merging it into the wrapping layout template html to provided the desired page look and content arrangement.
The Page Layout Module architecture supports flexibility in the content arrangement on the page by providing a template-based parameterized-controlled infrastructure for the pages layout definition.

Design Overview

Terms in use

JSP Template – A JSP file which defines a template for a page layout. Mainly contains HTMLB components and Layout TagLib components.

Page Builder - The module in charge of building the page's POM, fetching the iviews data and calling the Layout Component to construct the full page.

Layout Component - The main run-time engine for constructing page layout. A *JSP Template* based Portal Component that takes place in the page's POM.

Main Content Storage – Component that holds and manages iViews data (content, tray) fetched or created by *Page Builder. Layout Component* reads data from the component. aka "Main Storage Container".

Layout TagLib– (JSP) Tag Library in use within *JSP Templates* to serve *Layout Component* needs (i.e. - place holders for iViews Containers).

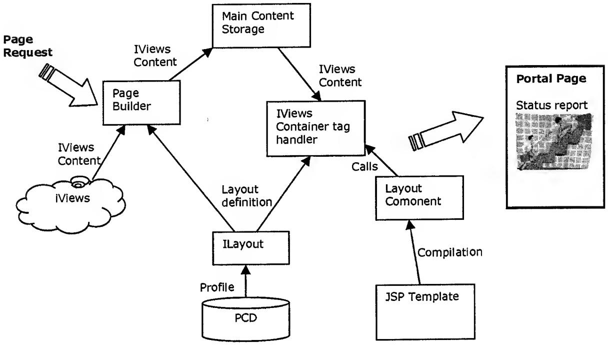
iViews Container – A page layout entity which contains iViews that should be presented together, in a certain order, on the page (i.e. - one column of iviews)

ILayoutStructure –Interface for getting page layout structure as containers and iviews arrangement on the page.

Architecture and Core Behaviors

General Architecture Description

Remark – This section describes the key players and the general concept in the page layout and html creation.



- A pre-defined *JSP Template* holds a general structure for the page layout (i.e. – template for pages with 3 vertical columns)
- The *Page Builder* gets a request for a certain page and builds the required POM, which includes the page's Iviews, and the *Layout Component,* which is an Abstract Portal Component, generated from the *JSP Template*.
- The Portal Component Profile for the *Layout Component* represents the page layout definitions as stored in the PCD.
- The *Page Builder* calls an *ILayoutStructure* object which translate the *Layout Component* profile to the requested semantic data (i.e. – the order the iviews should appear in the page).
- The *Page Builder* stores the iviews generated content in the *Main Content Storage* object.
- The *Layout Component* calls the *IViews Containers* tag handlers to include their iViews content in the necessary position in the page.
- Each tag handler calls the *ILayoutStructure* to get the list of the iViews it should include and then turns to the *Main Content Storage* to get the iViews data to include in the page.

The JSP Template

A JSP file is used as a layout template for the page rendering. The file contains mainly HTMLB components (using HTMLB TagLib) and page layout specific objects (using *Layout TagLib*). Each portal page is associated with one of the pre-defined JSP Templates to set it's layout pattern.
The EP will be provided with several Out-Of-The-Box JSP Templates (i.e. 2 columns page, 3 columns Page etc.). The customer will be able to create new JSP Templates to serve his unique layout (and page template) specifications.

At Run-Time the *JSP Template*, associated with the page, will be loaded by the PRT and added to the Page POM, as the *Layout Component*, under the *Page Builder*.

The approach of establishing the Page Layout mechanism upon JSP Templates provide several benefits:

- Flexibility – The template can be modified by the customer, thus creating any kind of layout template according to his needs (i.e. – a layout template of

  one row followed by three columns followed by another row).
- PRT supported – Being JSP based, the template is 'naturally' loaded and compiled as an AbstractPortalComponent and used in the Page POM.
- HTMLB – Using HTMLB TagLib enables basing the template upon HTMLB components thus enjoying HTMLB rendering and styling services.
- Devices – Optional approach to support different layouts for different devices can be using several device-oriented JSP Templates for the same page.

HTMLB Grid Layout component is used as a primary component to set the page layout structure.

The Layout TagLib

Defines tags to be used in the *JSP Templates* to serve layout purposes.
The module provide taglib definitions and tag-handlers implementation.

The **Container** Tag
A place-holder for *iViews Container*.
Attributes:
id – **unique** identifier for the container within this *JSP Template.*
orientation – either 'vertical' or 'horizontal'. Sets orientation for iViews arrangement within the container.
Syntax:
<lyt:container id="container1" orientation="vertical" />

JSP Template Code Example

Example for three-columns-layout JSP Template

```
<%@ taglib uri="prt:taglib:tlhtmlb" prefix="hbj" %>
<%@ taglib uri="prt:taglib:tllayout" prefix="lyt" %>

<hbj:content id="myContext" >
<hbj:page title="Portal Page">
<H1>Two Columns Layout</H1>
    <hbj:gridLayout   id="myGridLayout1"  width="100%">
        <hbj:gridLayoutCell rowIndex="1" columnIndex="1" width="50%"
verticalAlignment="top">
            <lyt:container id="column1"  orientation="vertical" />
        </hbj:gridLayoutCell>
        <hbj:gridLayoutCell rowIndex="1" columnIndex="2" width="50%"
verticalAlignment="top">
            <lyt:container id="column2"  orientation="vertical" />
        </hbj:gridLayoutCell>
    </hbj:gridLayout>
</hbj:page>
</hbj:content>
```

The Container Tag Handler

This Tag Handler writes the iViews' content to the page, according to their arrangement in the *iViews Container.*

Handling Sequence

1. Acquire reference to *ILayoutStructure* implementing object.
2. Get List of iViews in Container (call ILayoutStructure:getContainerIViews).
3. Acquire reference to *Main Content Storage* (IContentStorage implementation).
4. Create HTMLB GridLayout component.
5. For each iView in the List of iViews –
    a. Create HTMLB GridLayoutCell component.
    b. Retrieve iView's content .
    c. Set GridLayoutCell content to iView's content.
    d. Add GridLayoutCell to GridLayout.
6. Render GridLayout.

The Orientation Property

Property determines whether container should be presented as column (value = "vertical") or as row (value = "horizontal") of iviews.
For vertical orientation the GridLayout will have one column and many rows (as iViews number).
For horizontal orientation it will have one row and many columns.

Container Styling

Html styles for the container will be affected by the wrapping html styles. There will be no specific styling attributes for the *iViews Container* in the *JSP Template.*

Key APIs

Interface ILayoutStructure

public interface **ILayoutStructure**
Title: ILayoutStructure
Description: Interface for getting page layout structure as containers and iviews arrangement on the page.
Copyright (c) SAP Europe GmbH 2002

Method Summary

java.util.List	getContainerIViews(java.lang.String containerId) Get a list of iviews in a certain iViews Container, ordered by their rendering order on the page.
java.util.List	getPageIViews() Get a list of all iviews in the Page, ordered by their rendering order on the page.
java.util.List	getPageIViewsNoDuplicates() Get a list of all iviews in the Page, ordered by their rendering order on the page.

Method Detail

getContainerIViews

public java.util.List **getContainerIViews**(java.lang.String containerId)
Get a list of iviews in a certain iViews Container, ordered by their rendering order on the page.
Parameters:
containerId - The iViews Container ID.
Returns:
The IViews list.

getPageIViews

public java.util.List **getPageIViews**()
Get a list of all iviews in the Page, ordered by their rendering order on the page. Full iViews sequence will be returned, including IViews that apprears more than once in the page.
Returns:
The iViews List (including duplicates).

getPageIViewsNoDuplicates

public java.util.List **getPageIViewsNoDuplicates**()
Get a list of all iviews in the Page, ordered by their rendering order on the page. IViews sequence will be returned, but with no duplicates. each iView will appear only once in the returned list.
Returns:
The iViews List (no duplicates).

Remarks

- Provided the current PCD layout attributes structure, the implementing object might need to perform more initialisation actions as building a key-value mapping enumeration between container id in the *JSP Template* and container identifier in PCD.

Detailed Design

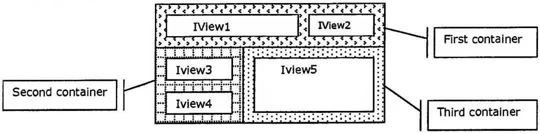
Layout PCD Structure

A Page context will have sub-context for each defined layout for the page. Each page will have one-primary layout (per device ?) which is the default layout presented to the user. The administrator can define alternative layouts for the user to choose from.

Each Layout sub-context holds the following sets of properties (distinguished by name-space):

- Iviews arrangement – stores iViews arrangement inside *iViews Containers.*
- General Properties – admin defined properties to be used in *JSP Template.*

Layout sample -



PCD Structure

- Layout1
  - Title = "myLayout"
    Containers arrangement
  - cont1.id = "myContainer1"
  - cont2.id = "myContainer2"
  - cont3.id = "myLastContainer"
  - cont1.title = "my Container 1"
  - cont2.title = "my Container 2"
  - cont3.title = "my Last Container"
  -

Mapping between containers sequence on the page and their id (as mentioned in the JSP Template) .
Attribute name is contX where X represent the sequence.

- Page1
  - Layout1

    IViews arrangement
    - cont1.ivu1 = "Iview1"
    - cont1.ivu2 = "Iview2"
    - cont2.ivu1 = "Iview3"
    - cont2.ivu2 = "Iview4"
    - cont3.ivu1 = "Iview5"

  - Layout2
    -
    -
  - Iview1
  - Iview2

Iviews arrangement inside containers.
Attribute name is contX_ivuY where contX is the container identifier and Y represents the ivu position in the iViews sequence in the container.

- Iview3
- Iview4
- Iview5

activeLayout = Layout1

Updated design -

```
<component name="pageLayout" >

    <component-profile>

      <property name="cont1" value="headerContainer">
         <property name="title" value="Header Area"/>
         <property name="ivu1" value="pageBuilder.iview"/>
         <property name="ivu2" value="pageBuilder.Iview"/>
      </property>
      <property name="cont2" value="navPanelContainer">
         <property name="title" value="Navigation Panel"/>
         <property name="ivu1" value="pageBuilder.iview"/>
         <property name="ivu2" value="pageBuilder.iview"/>
         <property name="ivu3" value="pageBuilder.iview"/>
      </property>
      <property name="cont3" value="workAreaContainer">
         <property name="title" value="Work Area"/>
         <property name="ivu1" value="pageBuilder.Iview"/>
      </property>
     </component-profile>
   </component>
```

remark: cont names represent sequence. Editing layout (jsp) and changing containers sequence must be followed by cont names update in PCD/Profile.

Open Issues

1. Frameset Support – special JSP Template ? actual page created ? special Layout Tags ?
2.